Usando GAP para trabalhar com códigos

Fábio Meneghetti

1 de outubro de 2019

GAP – Groups, Algorithms, Programming

- Um software livre e gratuito para álgebra discreta computacional
- `https://www.gap-system.org`
- Pacote Debian/Ubuntu: `sudo apt install gap-core`
- Iniciar a shell: `gap`
- `?comando` para instruções sobre um comando.

GAP – Groups, Algorithms, Programming

- Um software livre e gratuito para álgebra discreta computacional
- `https://www.gap-system.org`
- Pacote Debian/Ubuntu: `sudo apt install gap-core`
- Iniciar a shell: `gap`
- `?comando` para instruções sobre um comando.

GAP – Groups, Algorithms, Programming

- Um software livre e gratuito para álgebra discreta computacional
- `https://www.gap-system.org`
- Pacote Debian/Ubuntu: `sudo apt install gap-core`
- Iniciar a shell: `gap`
- `?comando` para instruções sobre um comando.

GAP – Groups, Algorithms, Programming

- Um software livre e gratuito para álgebra discreta computacional
- `https://www.gap-system.org`
- Pacote Debian/Ubuntu: `sudo apt install gap-core`
- Iniciar a shell: `gap`
- `?comando` para instruções sobre um comando.

GUAVA

- Uma biblioteca do GAP para trabalhar com códigos corretores de erros
- `https://gap-packages.github.io/guava`
- Pacote Debian/Ubuntu: `sudo apt install gap-guava`
- Importar a biblioteca: `LoadPackage("guava");`

GUAVA

- Uma biblioteca do GAP para trabalhar com códigos corretores de erros
- `https://gap-packages.github.io/guava`
- Pacote Debian/Ubuntu: `sudo apt install gap-guava`
- Importar a biblioteca: `LoadPackage("guava");`

GUAVA

- Uma biblioteca do GAP para trabalhar com códigos corretores de erros
- `https://gap-packages.github.io/guava`
- Pacote Debian/Ubuntu: `sudo apt install gap-guava`
- Importar a biblioteca: `LoadPackage("guava");`

GUAVA

- Uma biblioteca do GAP para trabalhar com códigos corretores de erros
- `https://gap-packages.github.io/guava`
- Pacote Debian/Ubuntu: `sudo apt install gap-guava`
- Importar a biblioteca: `LoadPackage("guava");`

Corpos Finitos

- O corpo finito com q elementos é denotado `GF(q)`.
- `Z(q)` é um gerador multiplicativo do corpo finito com q elementos. `0*Z(q)` é o elemento neutro aditivo, e `Z(q)^0` é o neutro multiplicativo.

Corpos Finitos

- O corpo finito com q elementos é denotado `GF(q)`.
- `Z(q)` é um gerador multiplicativo do corpo finito com q elementos. `0*Z(q)` é o elemento neutro aditivo, e `Z(q)^0` é o neutro multiplicativo.

Códigos e palavras

- Palavra $v = (1, 0, 2)$ em $\mathbb{F}_5^3$:

```
v := Codeword("102", GF(5));
```

- Código composto pelas palavras $(1, 0, 0)$ e $(1, 1, 1)$ em $\mathbb{F}_2^3$:

```
C := ElementsCode(["100", "111"], GF(2));
```

Códigos e palavras

- Palavra $v = (1, 0, 2)$ em $\mathbb{F}_5^3$:

```
v := Codeword("102", GF(5));
```

- Código composto pelas palavras $(1, 0, 0)$ e $(1, 1, 1)$ em $\mathbb{F}_2^3$:

```
C := ElementsCode(["100", "111"], GF(2));
```

Códigos lineares

- A matriz inteira

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \end{bmatrix},$$

  por exemplo, é denotada por

```
M := [[1,0,0],[0,2,1]];
```

- Código linear C gerado pela matriz M em $\mathbb{F}_q^n$:

```
C := GeneratorMatCode(M, GF(q));
```

Códigos lineares

- A matriz inteira

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \end{bmatrix},$$

  por exemplo, é denotada por

  ```
  M := [[1,0,0],[0,2,1]];
  ```

- Código linear C gerado pela matriz M em $\mathbb{F}_q^n$:

  ```
  C := GeneratorMatCode(M, GF(q));
  ```

Exemplos

Testar funções:

- `IsCode(C);`
- `IsLinearCode(C);`
- `List(C);`
- `MinimumDistance(C);`
- `RandomLinearCode(n,k,GF(q));`
- `DualCode(C);`

Código de Hadamard

- Uma matriz de Hadamard é uma matriz quadrada $H_{n \times n}$ com entradas em $\{-1, 1\}$ e tal que $H \cdot H^t = -nI$;
- Uma matriz de pode ser transformada numa matriz binária $A_{n \times n}$ trocando 1 por 0 e -1 por 1;
- Um código de Hadamard de **tipo 1** é gerado pelas linhas de A após se excluir a primeira coluna;
- Um código de Hadamard de **tipo 2** é criado adicionando os complementos das palavras já existentes;
- Um código de Hadamard de **tipo 3** é gerado pelas linhas de A e seus complementos.

Código de Hadamard

- Uma matriz de Hadamard é uma matriz quadrada $H_{n\times n}$ com entradas em $\{-1, 1\}$ e tal que $H \cdot H^t = -nI$;
- Uma matriz de pode ser transformada numa matriz binária $A_{n\times n}$ trocando 1 por 0 e -1 por 1;
- Um código de Hadamard de **tipo 1** é gerado pelas linhas de A após se excluir a primeira coluna;
- Um código de Hadamard de **tipo 2** é criado adicionando os complementos das palavras já existentes;
- Um código de Hadamard de **tipo 3** é gerado pelas linhas de A e seus complementos.

Código de Hadamard

- Uma matriz de Hadamard é uma matriz quadrada $H_{n \times n}$ com entradas em $\{-1, 1\}$ e tal que $H \cdot H^t = -nI$;
- Uma matriz de pode ser transformada numa matriz binária $A_{n \times n}$ trocando 1 por 0 e -1 por 1;
- Um código de Hadamard de **tipo 1** é gerado pelas linhas de A após se excluir a primeira coluna;
- Um código de Hadamard de **tipo 2** é criado adicionando os complementos das palavras já existentes;
- Um código de Hadamard de **tipo 3** é gerado pelas linhas de A e seus complementos.

Código de Hadamard

- Uma matriz de Hadamard é uma matriz quadrada $H_{n\times n}$ com entradas em $\{-1, 1\}$ e tal que $H \cdot H^t = -nI$;
- Uma matriz de pode ser transformada numa matriz binária $A_{n\times n}$ trocando 1 por 0 e -1 por 1;
- Um código de Hadamard de **tipo 1** é gerado pelas linhas de A após se excluir a primeira coluna;
- Um código de Hadamard de **tipo 2** é criado adicionando os complementos das palavras já existentes;
- Um código de Hadamard de **tipo 3** é gerado pelas linhas de A e seus complementos.

Código de Hadamard

- Uma matriz de Hadamard é uma matriz quadrada $H_{n\times n}$ com entradas em $\{-1, 1\}$ e tal que $H \cdot H^t = -nI$;
- Uma matriz de pode ser transformada numa matriz binária $A_{n\times n}$ trocando 1 por 0 e -1 por 1;
- Um código de Hadamard de **tipo 1** é gerado pelas linhas de A após se excluir a primeira coluna;
- Um código de Hadamard de **tipo 2** é criado adicionando os complementos das palavras já existentes;
- Um código de Hadamard de **tipo 3** é gerado pelas linhas de A e seus complementos.

Código de Hadamard

- O código de Hadamard gerado pela matriz de Hadamard H, de tipo t, é dado por `HadamardCode(H,t);`
- Também pode se usar $HadamardCode(n, t)$ que retorna um código de Hadamard com parâmetro n e tipo t.
- **Exemplo:** Testar com

`H := [[1,1,1,1],[1,-1,1,-1],[1,1,-1,-1],[1,-1,-1,1]];`

Código de Hadamard

- O código de Hadamard gerado pela matriz de Hadamard H, de tipo t, é dado por `HadamardCode(H,t);`
- Também pode se usar $HadamardCode(n,t)$ que retorna um código de Hadamard com parâmetro n e tipo t.
- **Exemplo:** Testar com

```
H := [[1,1,1,1],[1,-1,1,-1],[1,1,-1,-1],[1,-1,-1,1]];
```

Código de Hadamard

- O código de Hadamard gerado pela matriz de Hadamard H, de tipo t, é dado por `HadamardCode(H,t);`
- Também pode se usar $HadamardCode(n, t)$ que retorna um código de Hadamard com parâmetro n e tipo t.
- **Exemplo:** Testar com

```
H := [[1,1,1,1],[1,-1,1,-1],[1,1,-1,-1],[1,-1,-1,1]];
```

Códigos cíclicos

- `CyclicCodes(n,F)` lista todos os códigos cíclicos de tamanho n sobre o corpo F.
- `IsCyclicCode()`
- `BinaryGoppaCode()`, `TernaryGolayCode()`
- Gerando códigos cíclicos: um código cíclico pode ser visto como um ideal sobre $\mathbb{F}_q[X]/(X^n - 1)$.
- **Exemplo:** `x:= Indeterminate(GF(2));`

```
C1 := GeneratorPolCode(x^2+1, 5, GF(2));
C2 := GeneratorPolCode(x+1, 5, GF(2));
```

Códigos cíclicos

- `CyclicCodes(n,F)` lista todos os códigos cíclicos de tamanho n sobre o corpo F.
- `IsCyclicCode()`
- `BinaryGoppaCode()`, `TernaryGolayCode()`
- Gerando códigos cíclicos: um código cíclico pode ser visto como um ideal sobre $\mathbb{F}_q[X]/(X^n - 1)$.
- **Exemplo:** `x:= Indeterminate(GF(2));`

```
C1 := GeneratorPolCode(x^2+1, 5, GF(2));
C2 := GeneratorPolCode(x+1, 5, GF(2));
```

Códigos cíclicos

- `CyclicCodes(n,F)` lista todos os códigos cíclicos de tamanho n sobre o corpo F.
- `IsCyclicCode()`
- `BinaryGoppaCode()`, `TernaryGolayCode()`
- Gerando códigos cíclicos: um código cíclico pode ser visto como um ideal sobre $\mathbb{F}_q[X]/(X^n - 1)$.
- **Exemplo:** `x:= Indeterminate(GF(2));`
  `C1 := GeneratorPolCode(x^2+1, 5, GF(2));`
  `C2 := GeneratorPolCode(x+1, 5, GF(2));`

Códigos cíclicos

- `CyclicCodes(n,F)` lista todos os códigos cíclicos de tamanho n sobre o corpo F.
- `IsCyclicCode()`
- `BinaryGoppaCode()`, `TernaryGolayCode()`
- Gerando códigos cíclicos: um código cíclico pode ser visto como um ideal sobre $\mathbb{F}_q[X]/(X^n - 1)$.
- **Exemplo:** `x:= Indeterminate(GF(2));`

```
C1 := GeneratorPolCode(x^2+1, 5, GF(2));
C2 := GeneratorPolCode(x+1, 5, GF(2));
```

Códigos cíclicos

- `CyclicCodes(n,F)` lista todos os códigos cíclicos de tamanho n sobre o corpo F.
- `IsCyclicCode()`
- `BinaryGoppaCode()`, `TernaryGolayCode()`
- Gerando códigos cíclicos: um código cíclico pode ser visto como um ideal sobre $\mathbb{F}_q[X]/(X^n - 1)$.
- **Exemplo:** `x:= Indeterminate(GF(2));`

```
C1 := GeneratorPolCode(x^2+1, 5, GF(2));
C2 := GeneratorPolCode(x+1, 5, GF(2));
```

Código de Hamming

- `HammingCode(r,F)` retorna um código de Hamming com redundância r sobre o corpo F.
- **Exemplo:** `HammingCode( 3, GF(2) );`

Decodificação

- `Decode(C,v)` decodifica a palavra v usando o código C.
- `Decodeword(C,v)` diz a palavra do código C mais próxima de v.
- **Exemplo:**

```
C := HammingCode(3, GF(2));
v := Codeword("0111111", GF(2));
DistanceCodeword(v, Decodeword(C,v));
```

Decodificação

- `Decode(C,v)` decodifica a palavra v usando o código C.
- `Decodeword(C,v)` diz a palavra do código C mais próxima de v.
- **Exemplo:**

```
C := HammingCode(3, GF(2));
v := Codeword("0111111", GF(2));
DistanceCodeword(v, Decodeword(C,v));
```

Matrizes Geradora e Verificadora

- **Exemplo:** `C := RandomLinearCode(7,4,GF(3));`

  `G := GeneratorMat(C);`

  `H := CheckMat(C);`
- Usar `Display()` para visualizar as matrizes.

Outras coisinhas

- `ReedMullerCode()`
- `ReedSolomonCode()`
- `BCHCode()`
- Códigos algébrico-geométricos
- Ver mais no manual do GUAVA: `https://www.gap-system.org/Manuals/pkg/guava-3.14/doc/chap0.html`

Outras coisinhas

- `ReedMullerCode()`
- `ReedSolomonCode()`
- `BCHCode()`
- Códigos algébrico-geométricos
- Ver mais no manual do GUAVA: `https://www.gap-system.org/Manuals/pkg/guava-3.14/doc/chap0.html`

Outras coisinhas

- `ReedMullerCode()`
- `ReedSolomonCode()`
- `BCHCode()`
- Códigos algébrico-geométricos
- Ver mais no manual do GUAVA: `https://www.gap-system.org/Manuals/pkg/guava-3.14/doc/chap0.html`

Outras coisinhas

- `ReedMullerCode()`
- `ReedSolomonCode()`
- `BCHCode()`
- Códigos algébrico-geométricos
- Ver mais no manual do GUAVA: `https://www.gap-system.org/Manuals/pkg/guava-3.14/doc/chap0.html`

Outras coisinhas

- `ReedMullerCode()`
- `ReedSolomonCode()`
- `BCHCode()`
- Códigos algébrico-geométricos
- Ver mais no manual do GUAVA: `https://www.gap-system.org/Manuals/pkg/guava-3.14/doc/chap0.html`

Reticulados

- Seja G a matriz regular de uma forma bilinear simétrica (ex: no caso do produto interno é a matriz de Gram $G = B^t B$).
- Dado um inteiro $m \geq 0$, a função `ShortestVectors(G,m)` mostra os vetores inteiros x tais que $x \cdot G \cdot x^t \leq m$, e suas respectivas normas em G.
- Se G é a matriz de Gram, isso é equivalente a mostrar as coordenadas dos vetores menor norma do reticulado.
- **Exemplo:** `B := [[1,0,0], [1,1,0], [0,0,3]];`
  `G := TransposedMat(B)*B;`

Reticulados

- Seja G a matriz regular de uma forma bilinear simétrica (ex: no caso do produto interno é a matriz de Gram $G = B^t B$).
- Dado um inteiro $m \geq 0$, a função `ShortestVectors(G,m)` mostra os vetores inteiros x tais que $x \cdot G \cdot x^t \leq m$, e suas respectivas normas em G.
- Se G é a matriz de Gram, isso é equivalente a mostrar as coordenadas dos vetores menor norma do reticulado.
- **Exemplo:** `B := [[1,0,0], [1,1,0], [0,0,3]];`
  `G := TransposedMat(B)*B;`

Reticulados

- Seja G a matriz regular de uma forma bilinear simétrica (ex: no caso do produto interno é a matriz de Gram $G = B^t B$).
- Dado um inteiro $m \geq 0$, a função `ShortestVectors(G,m)` mostra os vetores inteiros x tais que $x \cdot G \cdot x^t \leq m$, e suas respectivas normas em G.
- Se G é a matriz de Gram, isso é equivalente a mostrar as coordenadas dos vetores menor norma do reticulado.
- **Exemplo:** `B := [[1,0,0], [1,1,0], [0,0,3]];`
  `G := TransposedMat(B)*B;`

Reticulados

- Seja G a matriz regular de uma forma bilinear simétrica (ex: no caso do produto interno é a matriz de Gram $G = B^t B$).
- Dado um inteiro $m \geq 0$, a função `ShortestVectors(G,m)` mostra os vetores inteiros x tais que $x \cdot G \cdot x^t \leq m$, e suas respectivas normas em G.
- Se G é a matriz de Gram, isso é equivalente a mostrar as coordenadas dos vetores menor norma do reticulado.
- **Exemplo:** `B := [[1,0,0], [1,1,0], [0,0,3]];`
  `G := TransposedMat(B)*B;`

Obrigado!

https://www.ime.unicamp.br/~ra155276/gap.pdf